OA-SLAM: Leveraging Objects for Camera Relocalization in Visual SLAM

Matthieu Zins, Gilles Simon, Marie-Odile Berger

OA-SLAM: Leveraging Objects for Camera Relocalization in Visual SLAM

Matthieu Zins* Gilles Simon* Marie-Odile Berger*

Université de Lorraine, Inria, LORIA, CNRS

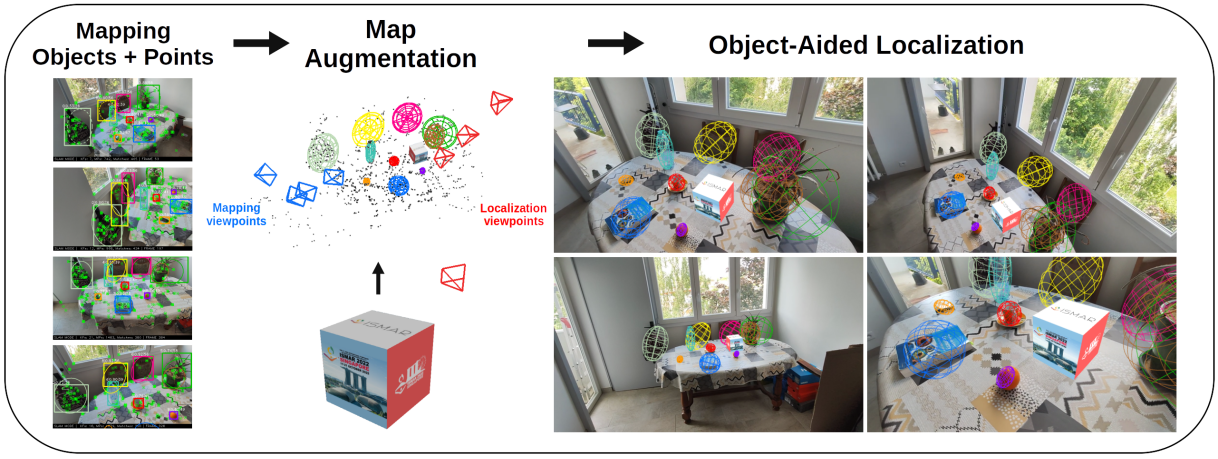

Figure 1: OA-SLAM enables on-the-fly object mapping and leverages them to improve the robustness of camera tracking reinitialization from a large variety of viewpoints. AR visualizations are shown on the right.

Abstract

In this work, we explore the use of objects in Simultaneous Localization and Mapping in unseen worlds and propose an object-aided system (OA-SLAM). More precisely, we show that, compared to low-level points, the major benefit of objects lies in their higher-level semantic and discriminating power. Points, on the contrary, have a better spatial localization accuracy than the generic coarse models used to represent objects (cuboid or ellipsoid). We show that combining points and objects is of great interest to address the problem of camera pose recovery. Our main contributions are: (1) we improve the relocalization ability of a SLAM system using high-level object landmarks; (2) we build an automatic system, capable of identifying, tracking and reconstructing objects with 3D ellipsoids; (3) we show that object-based localization can be used to reinitialize or resume camera tracking. Our fully automatic system allows on-the-fly object mapping and enhanced pose tracking recovery, which we think, can significantly benefit to the AR community. Our experiments show that the camera can be relocalized from viewpoints where classical methods fail. We demonstrate that this localization allows a SLAM system to continue working despite a tracking loss, which can happen frequently with an uninitiated user. Our code and test data are released at `gitlab.inria.fr/tangram/oa-slam`.

Index Terms: Computing methodologies—Computer graphics—Graphics systems and interfaces—Mixed / augmented reality; Computing methodologies—Artificial intelligence—Computer vision—Computer vision problems

*`forename.name@inria.fr`. This work was supported by the MoveOn project (Inria - DFKI).

1 Introduction

Augmenting the real world with virtual information to create interactive experiences has gained attention with a large potential in entertainment, serious games, medical training, retail, tourism industry or maintenance of complex equipment. For example, museums and cultural institutions have become very fond of such applications, in which they see new ways of engaging their visitors.

In order to visually enrich the real world with virtual information, high-precision and robust camera pose tracking is crucial. A certain understanding of the scene is also necessary for positioning the virtual elements, when building object-level interactive applications.

This is a well-known problem in computer vision and is called Simultaneous Localization and Mapping. It jointly builds a map of the environment and localizes the camera with respect to it. Many visual SLAM methods have been developed, turning RGB image sequences into sparse [7, 17, 23], semi-dense [8, 9], or dense [24] maps. RGB-D cameras have also been successfully used in SLAM systems [5, 6, 16, 31]. However, they are limited to medium-sized indoor environments and struggle with highly reflective scenes, such as metallic structures. Also, depth sensors are not yet widespread enough to be used in consumer applications, whereas all the recent smartphones have a decent RGB camera.

Despite decades of research, using SLAM for Augmented Reality is still challenging. Indeed, such applications are generally dedicated to the general public, including a majority of uninitiated users. In that context, fast or abrupt camera motions often occur. In addition, since the user is free of his/her motion, the system should be able to restart from any location in the scene. These challenging conditions are, however, generally not supported by existing camera tracking systems.

Current state-of-the-art SLAM methods, such as ORB-SLAM2 [23], rely on bag-of-words descriptors to find similar images and local appearance-based features, such as ORB or SIFT, to find matches between keypoints detected in the query image and landmarks in the map. However, using such low-level features is not well suited for relocalization when the viewing angle changes sig-

nificantly, because of their limited invariance to viewpoint changes and because some surfaces used to build the map may no longer be visible.

On the contrary, impressive progress have been made in the field of object detection over the last few years and deep learning-based techniques are now able to detect objects from a large variety of viewpoints and environmental conditions with great robustness. This naturally makes them good anchors to help visual-based camera localization.

In this work, we present a fully automatic object-aided (OA) SLAM system for unseen worlds, which is able to build a semantic map composed of 3D points and objects and leverages such high-level landmarks to improve its relocalization capability. Our main contributions are:

1. An improved relocalization method, combining the advantages of both objects and points, capable of estimating the camera pose from a large variety of viewpoints.

2. A fully automatic SLAM system capable of identifying, tracking and reconstructing objects, on the fly.

3. We demonstrate how our system can be used to reinitialize camera tracking on a previously built, and potentially augmented, map or to recover camera tracking after getting lost.

2 Related Work

We place ourselves in the context of SLAM in unseen world and with minimal effort for deployment. Methods with precise 3D models of objects, such as SLAM++ [29] and DeepSLAM++ [15], are thus not in the same line of work. For example, they require to have a database of CAD models of the objects seen during the deployment with specific networks training, which makes them painful to transplant into a new context. To be able to consider unseen environments, we are here interested in object-based systems which use more general kinds of modeling: cuboids [37], ellipsoids [25] or semantic point clouds [22, 33].

2.1 Object Mapping

Crocco proposed a closed-form formulation to estimate dual quadrics from multi-view object detections, using a simplified camera model in [4]. Rubino then extended it to the pinhole camera model [28]. In [3], Chen *et al*. addressed the problem of initial object estimation in the specific context of forward-translating camera movements, which commonly occurs in autonomous navigation. All these works are focused on object mapping and assume that the camera poses and object associations and provided.

2.2 Object-based Localization

Weinzaepfel *et al*. [35] proposed a method to compute the pose of the camera using dense 2D-3D correspondences between the objects present in a query image and those in reference images. However, this method is limited to planar objects.

More general objects, represented with ellipsoids, are used in [11] and [38]. However, compared to our work, these methods estimate the camera pose from objects only and assume a pre-built object map. Also, [11] only estimates the position of the camera and the orientation is assumed to be known. For its part, [38] is more focused on the improved 3D-aware elliptic detections of objects. They show how these better detections help to improve the accuracy of the estimated camera pose. However, the ellipse prediction network is trained on a specific scene, and thus, requires retraining in order to be used in a new scene, which is hardly acceptable for a SLAM system.

2.3 Object-based SLAM

A pioneering work, introducing objects in localization and mapping has been developed in [1], by Bao *et al*. , in which they recognize and localize objects within a Structure-from-Motion framework.

McCormac *et al*. [22] and Sünderhauf *et al*. [33] fused RGB-D SLAM with semantic segmentation and object detection to obtain semantically annotated dense point clouds. In such works the semantic information is added to the map after its creation but does not inform localization. Also, the created maps are not object-centric but dense point clouds where each point carries a semantic label.

In QuadricSLAM [25], Nicholson *et al*. derived a SLAM formulation which uses dual quadrics as 3D landmarks. They jointly estimate the camera pose and the dual quadrics parameters by combining odometry and high-level landmarks in factor graph-based SLAM. However, they assumed to have perfectly known data association, which considerably limits its application. EAO-SLAM [36] integrated objects in a semi-dense SLAM and exploited different statistics to improve the robustness of data association. Objects are represented as cuboids or ellipsoids, depending of their nature and are assumed to be placed parallel with the ground. This requires to know the vertical direction of the scene. Hosseinzadeh combined points, planes and quadrics into factor graph-based SLAM in [13] and [14]. They used an object detector network as well as a joint CNN to predict depth, surface normals and semantic segmentation, in order to capture the dominant structure of the scene and model point-plane, plane-plane and object-plane constraints. Their experiments show that incorporating objects and planes as new factors produce semantically meaningful maps and more accurate trajectories. In SO-SLAM [20], Liao *et al*. used manually extracted planes to add supporting constraints to objects, as well as, semantic scale priors and symmetry constraints.

An object-SLAM specialized for autonomous navigation was presented in [26]. They exploited bounding box detections, image texture, semantic knowledge and prior on objects shape (Toyota Camry) to infer ellipsoidal models and overcome the observability problem under forward-translating vehicle motions. However, only the object mapping part of their system is evaluated.

In CubeSLAM [37], Yang *et al*. used cuboids to represent objects. Their method is able to generate objects proposals from a single image using 2D bounding boxes and vanishing points sampling. The cuboids are then jointly optimized with camera poses and map landmarks. In [10], Frost *et al*. model objects with spheres and use them to resolve scale ambiguity and drift in SLAM. Recently, category-specific deep shape priors have been exploited for object reconstruction within a real-time SLAM system [34].

2.4 Object-based SLAM Relocalization

While the previously described works integrate objects into SLAM systems, mainly as new factors in the global optimization, the relocalization problem is never discussed. Most systems are based on ORB-SLAM2, and thus, rely on the classical point-based approach. Leveraging objects for pose recovery is precisely the real novelty of our work. Only Dudek *et al*. [19] leverage semantic mapping in SLAM for relocalization. However, their method is more a post-processing step which is able to estimate the similarity transform between two object maps created from two sequences.

Very recently, Mahattansin *et al*. improved relocalization in visual SLAM using object detections [21]. Compared to our work, no object map is reconstructed and object detections are only used to better filter keyframes candidates. The camera pose is still estimated using point matches with the most similar keyframe.

3 Motivations

Most of the existing object-based SLAM systems [14, 14, 20, 25, 26, 34] integrated objects in the core of a SLAM system through a joint camera-landmark-object optimization. However, [20] and the

monocular version of [34] noticed that it did not significantly improve the camera pose accuracy and [25] even observed a decreased accuracy compared to the point-based tracking of ORB-SLAM2. Objects represented with coarse models (cuboids or ellipsoids) might thus not be accurate enough to improve the camera pose tracking, but we nevertheless think that they are of great interest when the tracking gets lost.

Our fist motivation lies in the fact that current state-of-the-art object detectors reach a great robustness to viewpoint and illumination changes, which is very desirable for recovering the camera pose from a large variety of viewpoints. Also, to our knowledge, this way of using objects has not been explored a lot in existing object-based SLAM systems. Only [19] and [21] proposed a similar use of objects but they do not enable on-the-fly object map reconstruction and camera relocalisation.

Our second motivation was the lack of a fully automatic system for building object-oriented maps. Indeed, most existing methods require external information or manual processing. For example, QuadricSLAM [25] and SO-SLAM [20] assume that the problem of object association is solved. In their experiments, they manually annotated matches between the detected objects and the objects in the map. For their part, the authors of EAO-SLAM [36] assume that the objects are placed parallel with the ground, which necessitates to know the vertical direction of the scene. DSP-SLAM [34] is able to provide fine object reconstructions, but at the cost of training one additional shape prior network for each class of objects encountered during the use of the system. In contrast, we chose to use a generic and relatively coarse object model, while privileging the fully automatic aspect of our system.

4 Method

4.1 System Overview

Our system is detailed in Figure 2. It is based on ORB-SLAM2 (tracking, local mapping, loop closure) and augmented with additional modules dedicated to objects. These modules use the ellipse/ellipsoid modeling framework, described in section 4.2, and follow the same strategy as points, i.e. objects are tracked over frames, estimated in 3D, inserted into the map, and then, continuously refined. In particular, object tracking and object initial reconstruction are added to the main tracking thread. They are respectively described in sections 4.3 and 4.4. The local object mapping is handled in a similar way as the local point mapping and continuously refines the object models. It is run in a separate thread and is described in section 4.5. Finally, the relocalization module is enhanced by integrating objects, as described in section 4.6, greatly improving its robustness.

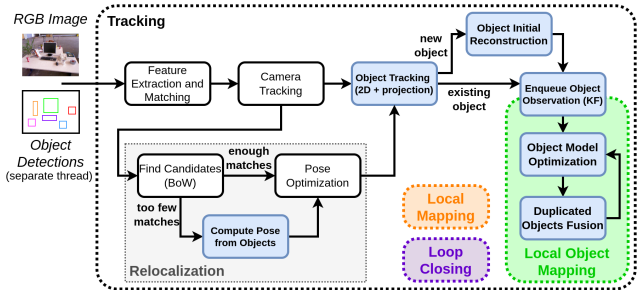

Figure 2: System: Blue items correspond to newly added elements within the ORB-SLAM2 backbone. Note that each module (Tracking, Local Mapping, Loop Closing and Local Object Mapping) is run in a separate thread.

4.2 Ellipsoidal Object Representation

In this work, we model an object with an ellipsoid, in 3D, and its observation in images with an ellipse. This is a coarse but lightweight representation which only requires nine parameters: three for its axes size, three for its orientation and three for its position. Also, an ellipsoid projects as an ellipse under any viewpoints, whose equation can be expressed in a closed-form manner using the dual space. In that space, the ellipsoid is defined by a 4×4 matrix Q^* and the ellipse by a 3×3 matrix C^*, which are linked together through a projection matrix P [12]:

$$C^* = PQ^*P^T. \tag{1}$$

In comparison, it would not be possible with a cuboidal representation of an object. Indeed, matching the 3D box corners with the 2D box edges leads to high combinatorics.

4.3 Object Detection and Association

We use the state-of-the-art object detection network YOLO [2] to obtain object detections in the video frames. Each detection includes an axis-aligned bounding box, a category and a detection score. For robustness, we only consider detections with a score higher than 0.5 and discard the others.

Establishing associations between object detections over time is a crucial part of our system. Given a set of detections in the current frame, the goal is to match each of them, either to an existing object track or decide to create a new one. Associations are firstly constrained by the object categories. We additionally take into account both the overlap of detection boxes as well as points matching between boxes. This allows us to handle inaccurate or partial object detections.

4.3.1 Box-based Object Tracking

Before being reconstructed, objects are tracked over frames in 2D, based on bounding boxes overlap and label consistency. This tracking is possible in the short term, when the motion between two frames is relatively small and smooth. However, it is prone to errors when detections are missing in some frames, when an object gets out of the camera field-of-view, or in case of abrupt camera motion.

Once 2D tracking of an object has been successfully performed on a sequence with a sufficient baseline, longer-term tracking can be obtained by considering its 3D reconstruction. For that, its ellipsoidal model is projected in the current frame (Eq. 1) and the overlap with the object detections in this frame is used to find associations.

In both cases, the optimal associations are found using the Hungarian algorithm [18], a well-known method for solving the assignment problem. This algorithm maximizes a total score of matching in order to find the best possible assignments between N detections and M objects. We define its score matrix at frame t as

$$\mathbf{S}^t = [s^t_{ij}]_{N \times M} \tag{2}$$

$$s^t_{ij} = \max(\text{IoU}(\mathtt{D}^t_i, \mathtt{B}^{t-k}_j), \underbrace{\text{IoU}(\mathtt{D}^t_i, \text{box}(\text{proj}(\mathtt{O}^t_j)))}_{\text{or 0 if reconstruction not yet available}}), \tag{3}$$

where $\mathtt{D}^t_i$ is the i-th detection in the current frame, $\mathtt{B}^{t-k}_j$ is the latest bounding box associated to the j-th object and $\mathtt{O}^t_j$ is the j-th object ellipsoid. The operations *IoU*, *box* and *proj* respectively stand for computing the intersection-over-union, the enclosing bounding-box and the projection in the current frame. The left IoU term refers to 2D bounding box tracking and the right one represents the long-term tracking by projection, only possible once an initial reconstruction of the object is available.

4.3.2 Point-based Object Tracking

The previously described long-term association method relies only on the geometry of the reconstructed ellipsoid, which is very interesting for small or texture-less objects, but can fail in case of truncated detections. Indeed, it is based on a global model of an object (ellipsoid), whose detection in the image may have some variance. Therefore, we additionally leverage points which provide a more precise 2D localization. During the camera pose estimation by the SLAM system, image keypoints are robustly matched to map landmarks. These matches can actually be used to link detections boxes and objects ellipsoids, using the following two statements:

- In the image, a keypoint is linked to a detection if it is situated inside the bounding box.
- In the map, a point-landmark is linked to an object if it is situated inside its ellipsoid.

If, at least, τ of such point-based matches exist, between a detection and an object, the association is considered. In our experiments, we used $\tau = 10$. This tracking approach is only enabled once the object has been reconstructed and is particularly useful for highly-textured objects or when the geometry-based tracking fails.

4.4 Initial Object Reconstruction

Our main idea is to generate 3D object hypotheses quickly after detection, in order that model based tracking can be performed in addition to box-based tracking. The validation and the integration of the object in the map comes later on, if tracking in subsequent frames is in coherence with the initial hypothesis. Otherwise, the object hypothesis is rejected.

Once an object has been successfully tracked over frames with a sufficient change in viewing angles, our system creates an initial rough estimate of its 3D ellipsoid. Before reconstructing, we check that the maximum angle between the rays passing through the camera center and the center of the object detections is above 10°. Rubino *et al*. proposed a method for reconstructing an ellipsoid from elliptic detections in images, however, we observed numerical instability for small baselines. In order to obtain a 3D estimate of an object as soon as possible, we opted for a simpler but more reliable approach, where an object is initially reconstructed as a sphere, and then, refined in the form of an ellipsoid as more views arrive. The position of this sphere is triangulated from the centers of the bounding boxes and its radius is determined as the mean bounding box size, back-projected at the estimated position, such that

$$radius = \frac{1}{2n}\sum_{i=1}^{n}\frac{z_i}{2}\left(\frac{w_i}{f_x}+\frac{h_i}{f_y}\right), \tag{4}$$

where z_i is the z-coordinate of the object center in the coordinate system of the i-th camera, w_i and h_i are the width and height of the detection box in the i-th frame, f_x and f_y are the camera intrinsics and n is the number of frames in which the object has been tracked.

The sphere is then refined as an ellipsoid (its orientation is set to identity) and its axes and position are updated when the object is detected in new images. This refinement is expressed in the form of a minimization of reprojection errors, similarly to the optimization described in the next section. Note that, from here, the data association can leverage this 3D estimate to search possible matches by projection. Once the object has been reconstructed and refined in a sufficient number of frames (typically 40 frames), the object is integrated in the map provided that the overlap between its projections and detections in all the frames where the object was tracked is sufficient. We set the minimal overlap threshold at 0.3.

It should be noted that, although we use relatively low thresholds to favor object tracking, our reconstruction method is robust both to false object detection and to false associations. Indeed, false-positive detections that YOLO might generate are generally not stable enough over time to be tracked. In addition, the requirement of a minimal overlap in all the frames in which an object is detected ensures that only coherent objects are integrated in the map.

4.5 Local Object Mapping

4.5.1 Objects Refinement

Similarly to point-landmarks in the local bundle adjustment of ORB-SLAM2, object models are also regularly refined. Each time a new keyframe observes an object present in the map, this object is updated through the minimization of a reprojection error. We use the Wasserstein distance between the ellipse inscribed inside the detection box and the projection of the estimated ellipsoid as objective function. This distance has already been used as cost between ellipses to train an ellipse predicting neural network in [27]. In this distance, ellipses are interpreted as 2D Gaussian distributions $\mathcal{N}(\mu,\Sigma)$ such that

$$\mu = \begin{bmatrix} c_x \\ c_y \end{bmatrix},\ \Sigma^{-1} = R(\theta)\begin{bmatrix} \frac{1}{\alpha^2} & 0 \\ 0 & \frac{1}{\beta^2} \end{bmatrix}R(\theta)^T, \tag{5}$$

where $[c_x, c_y]$ are the ellipse the center, $[\alpha,\beta]$ are its axes length and θ is its orientation. The Wasserstein distance between two Gaussian distributions $\mathcal{N}_1(\mu_1,\Sigma_1)$ and $\mathcal{N}_2(\mu_2,\Sigma_2)$ is computed by

$$\begin{aligned}\mathcal{W}_2^2(\mathcal{N}_1,\mathcal{N}_2) = &\|\mu_1-\mu_2\|_2^2 \\ &+ Tr(\Sigma_1+\Sigma_2-2(\Sigma_1^{\frac{1}{2}}\Sigma_2\Sigma_1^{\frac{1}{2}})^{\frac{1}{2}}).\end{aligned} \tag{6}$$

The ellipsoidal shape of the i-th object is thus refined by minimizing

$$\sum_{j=0}^{N}\sigma_j^{-1}\mathcal{W}_2^2(E_{ij}, P_j Q_i^* P_j^T), \tag{7}$$

where E_{ij} is the ellipse inscribed in the j-th detection, Q_i^* is the dual matrix of the i-th object, P_j is the projection matrix of the j-th keyframe, σ_j is the score of the detection in the j-th keyframe and N is the number of observations of the object.

4.5.2 Objects Fusion

In some cases, an object might be duplicated in the map. This can happen when a detected object was not visible for a few frames and data association fails to correctly re-match it with the existing track and inserts a new object in the map. To prevent such cases, our system regularly checks for duplicated objects. Two objects of the same category are considered as a unique object if their 3D aligned boxes IoU, which is very fast to compute, exceeds a certain threshold (0.2 in ours experiments), if the center of one ellipsoid lies inside the other one, or if they share more than τ common 3D landmarks. In such a case, the detection boxes tracked for both objects in keyframes are combined and a new ellipsoid is initialized, following the procedure described in section 4.4, but only on keyframes.

4.6 Relocalization using Objects

The original relocalization method of ORB-SLAM2 offers a good reliability but often fails when the query image is far from the past camera trajectory. Indeed, it uses Bag-of-Word descriptors to find similar keyframe candidates and searches for point matches, which fails frequently when the viewpoint on the reconstructed map differs significantly from the keyframe. As described in Figure 2, we enhance the relocalization with an object-based approach, more robust to viewpoint changes, which is triggered when too few point correspondences have been established. Indeed, objects learnt from large database have the advantage that they can be detected from a

large variety of viewpoints (front, back, top, side, ...), thus opening the way towards relocalization from any position without specific knowledge of the objects in the scene.

Knowing that poses computed from Perspective-n-Point (PnP) are more accurate than those obtained from object correspondences, our main idea is to guide point matching with the pose computed from 2D/3D object correspondences. This pose is generally sufficient so that projection of the landmark points are similar to detected points, thus allowing to perform easily point correspondences and to use PnP for localization.

Our object-based approach is a modified version of the algorithm presented in [38], which jointly determines object correspondences between the query image and the map and estimates the pose of the camera. Pairs of ellipse-ellipsoid are established based on their categories. At each iteration, a minimal set of three pairs are selected and the camera pose is computed with the Perspective-3-Point (P3P) algorithm on the ellipses and ellipsoids centers. P3P provides four potential solutions. For each pose, ellipsoids are projected and associated to detections based on their overlap. A cost is calculated as the sum of $1 - IoU$ for each associated pairs and the pose with the minimal cost among the four P3P solutions is selected. These poses are used to identify keypoint-landmark correspondences through the local matching step of ORB-SLAM2. Finally, the pose with the minimal cost and with more than 30 keypoint-landmark matches is selected and refined on points. The SLAM system can then resume tracking. If keypoint-landmark correspondences could not be established, for example in a texture-less environment, the pose with the minimal cost is selected, without refinement. In that case, the current camera pose is estimated, but tracking can not be recovered and the system will trigger a new relocalization procedure when the next image arrives.

5 Experiments and Results

5.1 Scenes and Objects

To evaluate our approach, we used two scenes from TUM RGB-D dataset [32]: *fr2/desk* and *fr3/long_office_household*. This dataset provides ground truth poses for the camera trajectory. However, it is limited to one scan per scene with a mostly orbital camera trajectory.

We thus also recorded our own sequences using a standard smartphone camera. This allowed us to evaluate our method in more diverse environments (Fig. 10) and from a larger variety of viewpoints, in terms of both angle and scale (Fig. 7). We used COLMAP [30] to obtain ground truth camera poses.

It also allowed us to use both very general objects (book, chair, cup, ...), but also more specific ones (statue, amphora, ...). The *sink* and *desk* scenes in Figure 10 show, for example, how our system can be used in an everyday life environment, using an off-the-shelf object detector. More specific objects, in particular texture-less statues, have also been tested (see Figures 7 and 10). This gives a good insight of how our system can be used for AR applications in museums, for example. For these objects, YOLO has been fine-tuned on a few manually annotated images (around 50 images).

The full videos of the experiments are available in the accompanying video.

5.2 Object Mapping

In this section, we first evaluate the object mapping part our system, which builds ellipsoidal models of the objects in the scene, on the fly, using the computed camera poses and object detections. Figure 3 shows the reconstructed map on *fr2/desk* and *fr3/long_office_household* sequences. We can observe that our coarse object models are globally well positioned on the objects in the map.

Figure 4 shows a small but challenging scene, where three duplicated objects are placed side by side. Indeed, object classes can not be used to constraint data association and the objects are occluded when seen from the side. Our system still manages to build three accurate ellipsoidal models.

Finally, we compare our reconstructed map with the one built with EAO-SLAM [36] in Figure 5. We used the code they released at: `https://github.com/yanmin-wu/EAO-SLAM`. The scenario is simple with objects placed on a table and an orbiting camera trajectory. The same object detections were passed to both methods. For EAO-SLAM, we had to manually set the vertical direction so that objects can be estimated parallel to the ground. Both methods are based on ORB-SLAM2 and reconstruct similar sparse point clouds. In terms of objects, our method reconstructs precise ellipsoidal models of all the objects present on the table (the relatively large laptop, the thin bottles and the small mouse). EAO-SLAM makes a distinction between objects with a regular or non-regular shape and represents them respectively with cuboids and ellipsoids. We can observe two cuboids created over the mouse and the black book (the red ellipsoid on the right side of the map) which should not exist. The bottles are represented with thicker ellipsoids which fit less well their shape. Finally, the mug placed behind the laptop has a much larger ellipsoidal reconstruction (the green ellipsoid in the middle in EAO's map and the small light-blue ellipsoid in our map).

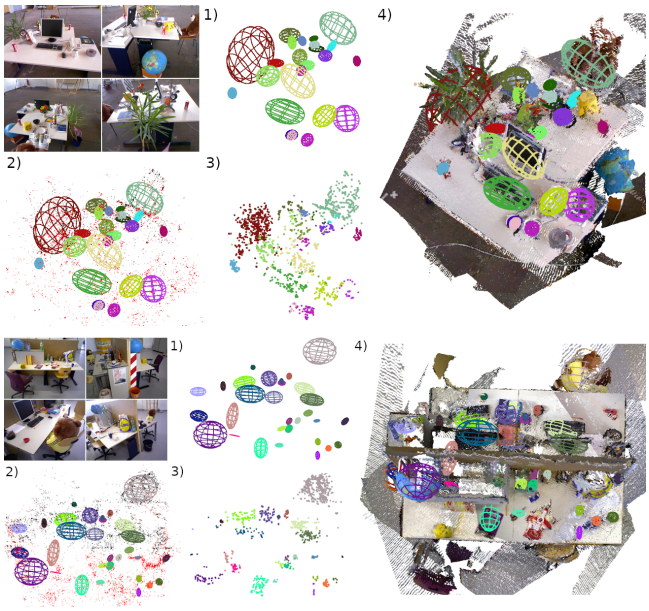

Figure 3: Obtained map on *fr2/desk* (top) and *fr3/long_office_household* (bottom). Upper-left images give an overview of the video sequences. 1) Object map; 2) Object and point map; 3) Points associated to objects; 4) Object map displayed over a dense reconstruction of the scene (only for illustration).

Figure 4: Obtained map for duplicated objects placed side-by-side. The images on the left give an overview of the sequence.

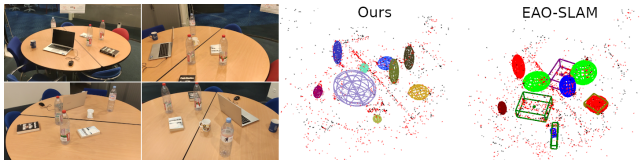

Figure 5: Comparison of the obtained maps using our method vs. EAO-SLAM [36]. The semi-dense map created by EAO-SLAM is not shown here, as they only use it for visualization.

5.3 Objects versus Points

5.3.1 Relocalization

In this section, we analyze the benefits of combining objects and points for relocalization. The scenario of the experiment is the following: we first map the scene from limited viewpoints, mostly from one side, with our SLAM system and then call the relocalization procedure on query images from different viewpoints. For *fr2/desk*, we split the sequence in two parts: the first 700 frames were used for mapping while the rest is used for relocalization. We also recorded our own sequences enabling more diverse viewpoints, in terms of angle and scale.

Figures 6 and 7 compare our estimated camera positions with the results obtained using the classical point-based method available in ORB-SLAM2. We can clearly see that our method is able to accurately localize the camera all around the scene. This is achieved thanks to the robustness of object detectors, which can detect objects even from behind. The *tour* and *near-far* test sequences in Figure 7 significantly varies the distance from the scene. The accuracy of the estimated poses only slightly decrease for query images taken very far or very close to the scene. On the contrary, the original point-based relocalization is only able to produce pose estimates for viewpoints close to the ones used for mapping. This is also visible on the curves in Figure 8, which show that a much larger proportion of images can be localized. Our method does not improve the accuracy of the point-based approach, mainly due to the coarse object representation, but significantly enlarges its operating area. The plateau at around 70% of localized images with our method on *fr2/desk* can be explained by the fact that no or only one object is visible in a part of the frames. We also evaluated an object-only approach on *fr2/desk*. The results in Figure 8 show that it keeps the robustness of objects but loses the accuracy of points (see the bottom left part of the curve).

5.3.2 Integrating Objects in Bundle Adjustment

In the previous experiments, we showed how objects are reconstructed by our SLAM system and why reasoning about such higher-level landmarks is beneficial for relocalization. In this section, we explore the question of using objects during camera tracking. We recall that, in our system, objects are not involved in camera tracking. They are refined independently in a separate optimization and are used for relocalization only. We thus created two other versions involving objects in the bundle adjustment (see Figure 9). A first one, called *Obj_dets*, where objects are integrated in the bundle adjustment without updating their ellipsoidal models. And a second one, called *Full_BA*, in which the objects models are completely integrated inside the bundle adjustment, together with camera poses and point-landmarks. The difficulty of combining point-based and object-based factors is that their cost need to be balanced. We thus arbitrarily scaled the object residuals in order to have values of the same order of magnitude as point residuals.

We measured the error of the estimated positions of keyframes on the fr2/desk and fr3/long_office_household sequences for different configurations of the bundle adjustment. The results, available in Table 1, do not show a clear improvement by integrating objects in the graph-based optimization, and even decrease the pose accuracy in the case of the full camera-point-object bundle adjustment. Note, that we did not compare with ORB-SLAM2 because, in such scenarios of pure camera pose tracking, it is equivalent to our method, using the first configuration.

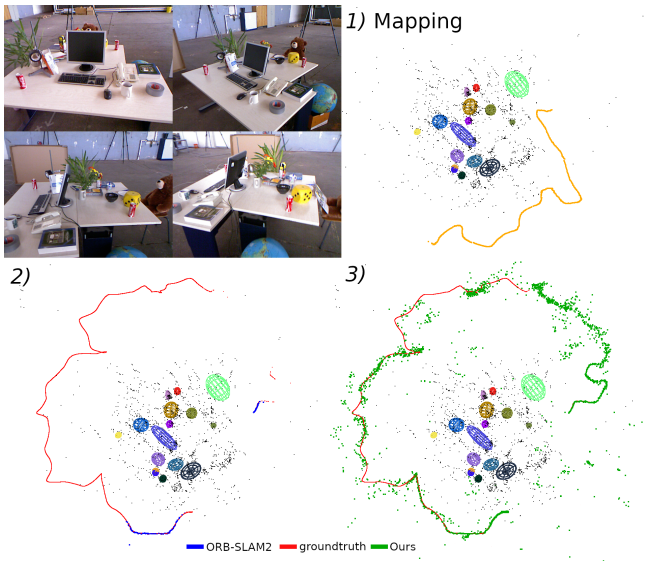

Figure 6: Estimated camera positions by the relocalization module (frame-by-frame) on *fr2/desk*. The top-left images give an overview of the frames used for mapping. 1) The orange camera trajectory was used for mapping; 2) Relocalized camera using ORB-SLAM2 in blue; 3) Relocalized camera using our system in green. The ground truth trajectory is in red.

We believe that coarse object models (ellipsoids) combined with simplified detections (axis-aligned bounding boxes), are not able to provide the same level of accuracy as current state-of-the-art point-based tracking in sufficiently textured areas. Similar results were notably obtained by [25] and [20], whereas [14] noted slight improvements.

Sequence	OA-SLAM (ours)	Obj_dets	Full_BA
fr2/desk	**0.808**	0.901	0.860
fr3/long_office_household	**1.180**	1.978	2.143

Table 1: RMSE (cm) of keyframes Absolute Trajectory Error (median value obtained on 20 runs).

5.4 Time Analysis

The speed of our system depends on the number of objects present in the scene and their diversity of category. The tracking part takes a median time of 27 ms per frame, only slightly more than ORB-SLAM2 which takes 25 ms. Our object-aided relocalization takes a median time of 22 ms per frame. We measured these times on the scene shown in Figure 7 using an Intel Xeon 3.6GHz CPU. These durations are totally satisfactory for the targeted applications. Also, note that this scene includes a relatively high number of objects (11), with in particular, five objects of the same type (the statues) which increases the number of potential object associations.

5.5 Application to AR

We demonstrate here two direct applications of the improved relocalization capabilities. Scenes of everyday life, in a kitchen or

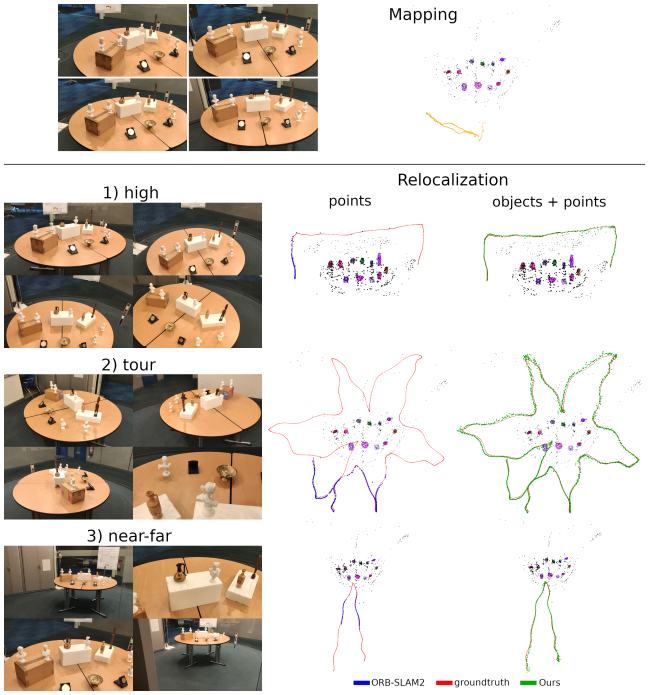

Figure 7: Estimated camera positions by the relocalization module (frame-by-frame) on a custom scene with a large viewpoints variety. Top: an overview of frames used for mapping and the obtained map with the estimated camera trajectory in orange. Bottom: relocalization results obtained on 3 test sequences. The images give an overview of the query frames. Camera positions estimated with ORB-SLAM2 are in blue, with our method in green and the groundtruth in red.

in a dining room, are considered in this section. These scenes are complex since the notion of objects is less obvious than in the previous examples, especially for the sink. The term "object" is used in a broad sense. It refers to any element of the scene that could be detected using a specifically trained object detector network.

5.5.1 Reinitializing 3D Tracking

The enhanced relocalization capability is particularly interesting to initialize camera 3D tracking in AR applications. Once a map of the working area has been built, and potentially augmented with virtual elements, the camera pose is registered with the map. We show, in Figures 1 and 10, how our method can be used for relatively complex scenarios, where the scene is mainly seen from one side, at a constant distance, during mapping and then localization is performed from the other side at varying distances.

5.5.2 SLAM Resume

A typical scenario is presented in Figure 11, which demonstrates the stronger recovery capacity of our SLAM system: (1-4) The system initially tracks the camera in 3D and builds a map of points and objects. (5-6) The tracking gets lost due to an abrupt camera motion (in this experiment the camera sees only the floor). (7-9) When the reconstructed scene is visible again, the relocalization module estimates the camera pose from objects, establishes point matches and enables the tracking and mapping to continue. In such a scenario, our object-based approach significantly extends the range of viewpoints from where SLAM recovery is possible, which is limited with the classical point-based method.

In both scenarios, ORB-SLAM2 fails to reinitialize or recover camera tracking during a large number of frames. This is illustrated in the accompanying video.

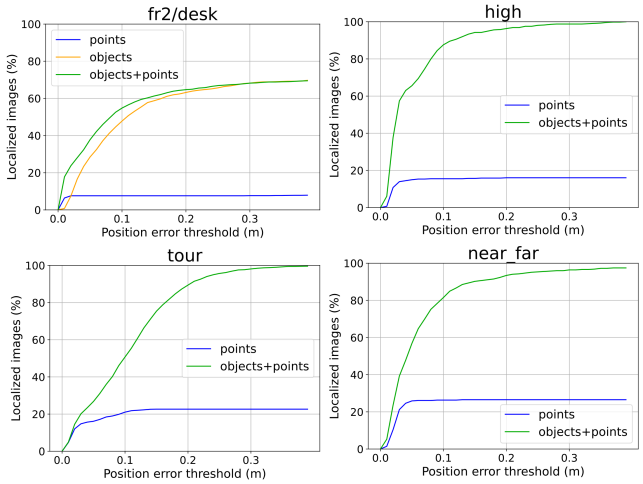

Figure 8: Percentage of estimated camera positions with respect to an error threshold on the position.

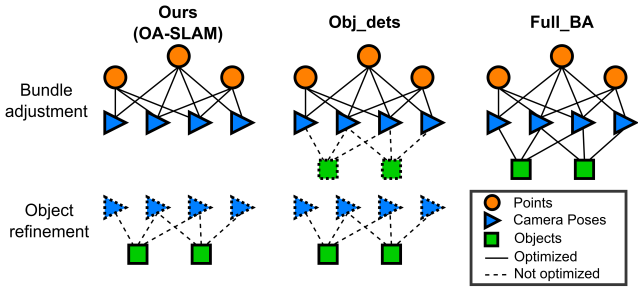

Figure 9: Three configurations for integrating objects in the SLAM bundle adjustment.

5.6 By-Part Modeling

Figure 12: Illustration of by-part modeling. Left: reconstructed map. Right: localized images using complete objects (bottom row) or parts (top row).

Depending on the context, it can be useful to adapt the level of details of the scene modeling (full object vs. parts). When the camera sees the scene from relatively far, objects appear small in the image but are generally in a sufficient number to allow relocalization, which requires at least three objects. In such cases, using global ellipsoidal models is sufficient. However, when the camera gets closer, only one or two objects may be visible and a by-part modeling becomes particularly desirable to increase the number of potential anchors. It is also interesting to better handle objects which are partially occluded.

Our method can handle this in a very flexible way, as the detector network can be fine-tuned to detect distinguishable parts of an object, only requiring some manual annotations in a few images. Figure 12 shows the results of our method, after fine-tuning YOLO to detect

Figure 10: Camera tracking reinitialization on three scenes (*sink*, *desk*, *museum*). For each scene, an overview of the frames used for mapping are shown in the top-left corner and the four frames below correspond to query images localized with our system.

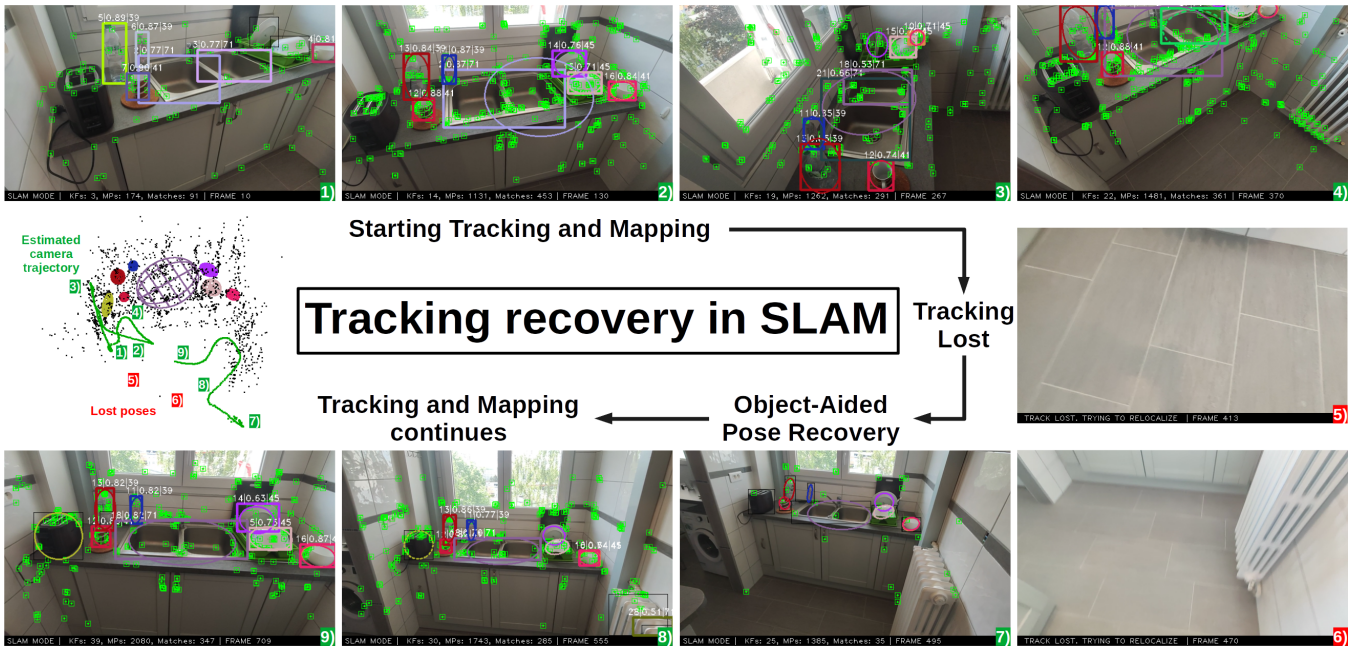

Figure 11: Camera tracking recovery with OA-SLAM. The numbers displayed above the object detections in the images are respectively the id of their associated object, their detection score and their class.

parts of the statue (head, shoulders and bottom). In the case of a close camera (top row), these parts are used for relocalization, whereas only full object detections are used when the camera is far from the scene (bottom row).

5.7 Discussion and Conclusion

In this paper we proposed to integrate objects to point-based monocular SLAM and use them as higher-level landmarks to improve its relocalization ability. Our system leverages existing object detection networks and is able to build a lightweight object map, on the fly.

Comparing our method to the state-of-the-art ORB-SLAM2 system, we show that, thanks to objects, our system is able to relocalize from a significantly larger variety of viewpoints. We demonstrate through experiments that this improved relocalization can be used for initializing 3D tracking on an augmented map or for SLAM tracking recovery after getting lost in the context of AR application. The coarse ellipsoidal models used for objects allow us to reach our objectives of deployment in an unseen world with minimal effort.

While we demonstrated the efficiency of our system in our experiments, it has also some limitations. First, our relocalization approach requires that, at least, three objects present in the map are detected in the query image. This requirement can limit its efficiency, especially in case of a very tight field-of-view, but it can be reduced using a by-part modeling of objects. Secondly, dynamic objects in the scene are not handled by our system. For that, a combination of our object-based reasoning with static/dynamic identification can be considered in future works.

References

[1] S. Y. Bao, M. Bagra, Y. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pp. 2703–2710. IEEE Computer Society, 2012. doi: 10.1109/CVPR.2012.6247992

[2] A. Bochkovskiy, C. Wang, and H. M. Liao. Yolov4: Optimal speed and accuracy of object detection. *CoRR*, 2020.

[3] S. Chen, S. Song, J. Zhao, T. Feng, C. Ye, L. Xiong, and D. Li. Robust dual quadric initialization for forward-translating camera movements.

IEEE Robotics Autom. Lett., 6(3):4712–4719, 2021. doi: 10.1109/LRA.2021.3067868

[4] M. Crocco, C. Rubino, and A. D. Bue. Structure from motion with objects. In *CVPR*, 2016.

[5] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph.*, 36(3):24:1–24:18, 2017. doi: 10.1145/3054739

[6] F. Endres, J. Hess, J. Sturm, D. Cremers, and W. Burgard. 3-d mapping with an rgb-d camera. *IEEE Transactions on Robotics*, 30(1):177–187, 2014. doi: 10.1109/TRO.2013.2279412

[7] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. In *arXiv:1607.02565*, July 2016.

[8] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: large-scale direct monocular SLAM. In D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds., *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part II*, vol. 8690 of *Lecture Notes in Computer Science*, pp. 834–849. Springer, 2014.

[9] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.

[10] D. P. Frost, V. A. Prisacariu, and D. W. Murray. Recovering stable scale in monocular SLAM using object-supplemented bundle adjustment. *IEEE Trans. Robotics*, 34(3):736–747, 2018. doi: 10.1109/TRO.2018.2820722

[11] V. Gaudillière, G. Simon, and M.-O. Berger. Camera Relocalization with Ellipsoidal Abstraction of Objects. In *ISMAR 2019 - 18th IEEE International Symposium on Mixed and Augmented Reality*, pp. 19–29. Beijing, China, Oct. 2019.

[12] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second ed., 2004.

[13] M. Hosseinzadeh, Y. Latif, T. Pham, N. Sünderhauf, and I. D. Reid. Structure aware SLAM using quadrics and planes. In *ACCV*, 2018.

[14] M. Hosseinzadeh, K. Li, Y. Latif, and I. D. Reid. Real-time monocular object-model aware sparse SLAM. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pp. 7123–7129. IEEE, 2019. doi: 10.1109/ICRA.2019.8793728

[15] L. Hu, W. Xu, K. Huang, and L. Kneip. Deep-slam++: Object-level RGBD SLAM based on class-specific deep shape priors. *CoRR*, 2019.

[16] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. A. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. J. Davison, and A. W. Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In J. S. Pierce, M. Agrawala, and S. R. Klemmer, eds., *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology, Santa Barbara, CA, USA, October 16-19, 2011*, pp. 559–568. ACM, 2011. doi: 10.1145/2047196.2047270

[17] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *Sixth IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2007, 13-16 November 2007, Nara, Japan*, pp. 225–234. IEEE Computer Society, 2007.

[18] H. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistic Quarterly*, 2, 05 2012.

[19] J. Li, D. Meger, and G. Dudek. Semantic mapping for view-invariant relocalization. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pp. 7108–7115. IEEE, 2019. doi: 10.1109/ICRA.2019.8793624

[20] Z. Liao, Y. Hu, J. Zhang, X. Qi, X. Zhang, and W. Wang. SO-SLAM: semantic object SLAM with scale proportional and symmetrical texture constraints. *IEEE Robotics Autom. Lett.*, 7(2):4008–4015, 2022. doi: 10.1109/LRA.2022.3148465

[21] N. Mahattansin, K. Sukvichai, P. Bunnun, and T. Isshiki. Improving relocalization in visual slam by using object detection. In *2022 19th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, pp. 1–4, 2022. doi: 10.1109/ECTI-CON54298.2022.9795637

[22] J. McCormac, A. Handa, A. J. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation, ICRA 2017, Singapore, Singapore, May 29 - June 3, 2017*, pp. 4628–4635. IEEE, 2017. doi: 10.1109/ICRA.2017.7989538

[23] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. *IEEE Trans. Robotics*, 33(5):1255–1262, 2017. doi: 10.1109/TRO.2017.2705103

[24] R. A. Newcombe, S. Lovegrove, and A. J. Davison. DTAM: dense tracking and mapping in real-time. In D. N. Metaxas, L. Quan, A. Sanfeliu, and L. V. Gool, eds., *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pp. 2320–2327. IEEE Computer Society, 2011. doi: 10.1109/ICCV.2011.6126513

[25] L. Nicholson, M. Milford, and N. Sünderhauf. Quadricslam: Dual quadrics from object detections as landmarks in object-oriented SLAM. *IEEE Robotics Autom. Lett.*, 2019.

[26] K. Ok, K. Liu, K. M. Frey, J. P. How, and N. Roy. Robust object-based SLAM for high-speed autonomous navigation. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pp. 669–675. IEEE, 2019. doi: 10.1109/ICRA.2019.8794344

[27] S. Pan, S. Fan, S. W. K. Wong, J. V. Zidek, and H. Rhodin. Ellipse detection and localization with applications to knots in sawn lumber images. In *WACV*, 2021.

[28] C. Rubino, M. Crocco, and A. D. Bue. 3d object localisation from multi-view image detections. *IEEE TPAMI*, 2018.

[29] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *CVPR*, 2013.

[30] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[31] T. Schöps, T. Sattler, and M. Pollefeys. BAD SLAM: bundle adjusted direct RGB-D SLAM. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pp. 134–144. Computer Vision Foundation / IEEE, 2019. doi: 10.1109/CVPR.2019.00022

[32] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[33] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. D. Reid. Meaningful maps with object-oriented semantic mapping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2017, Vancouver, BC, Canada, September 24-28, 2017*, pp. 5079–5085. IEEE, 2017. doi: 10.1109/IROS.2017.8206392

[34] J. Wang, M. Rünz, and L. Agapito. DSP-SLAM: object oriented SLAM with deep shape priors. In *International Conference on 3D Vision, 3DV 2021, London, United Kingdom, December 1-3, 2021*, pp. 1362–1371. IEEE, 2021. doi: 10.1109/3DV53792.2021.00143

[35] P. Weinzaepfel, G. Csurka, Y. Cabon, and M. Humenberger. Visual localization by learning objects-of-interest dense match regression. In *CVPR*, 2019.

[36] Y. Wu, Y. Zhang, D. Zhu, Y. Feng, S. Coleman, and D. Kerr. EAO-SLAM: monocular semi-dense object SLAM based on ensemble data association. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2020, Las Vegas, NV, USA, October 24, 2020 - January 24, 2021*, pp. 4966–4973. IEEE, 2020. doi: 10.1109/IROS45743.2020.9341757

[37] S. Yang and S. A. Scherer. Cubeslam: Monocular 3-d object SLAM. *IEEE Trans. Robotics*, 2019.

[38] M. Zins, G. Simon, and M. Berger. Object-based visual camera pose estimation from ellipsoidal model and 3d-aware ellipse prediction. *Int. J. Comput. Vis.*, 130(4):1107–1126, 2022. doi: 10.1007/s11263-022-01585-w